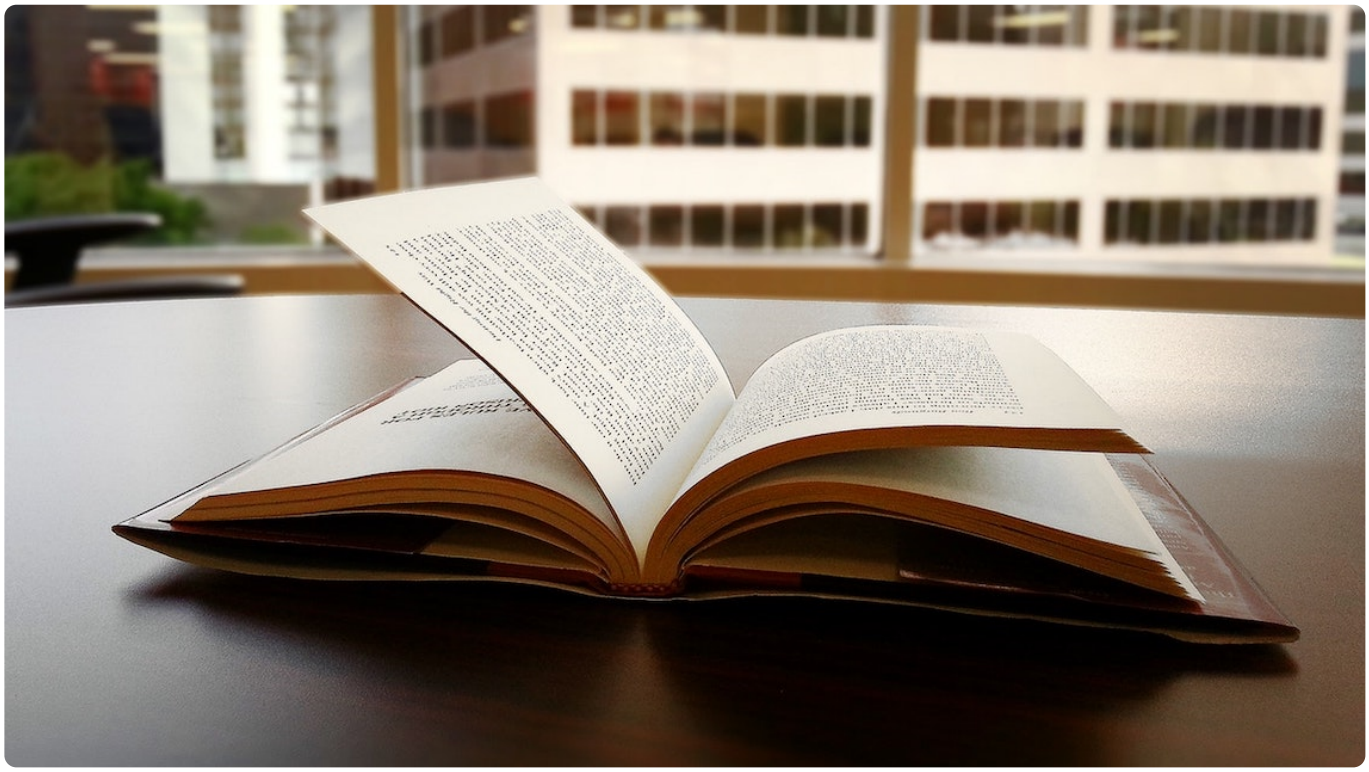
10 | 性能测试的规划和步骤：为什么性能测试不容易一蹴而就呢？

2019-12-18 庄振运

性能工程高手课 进入课程 >

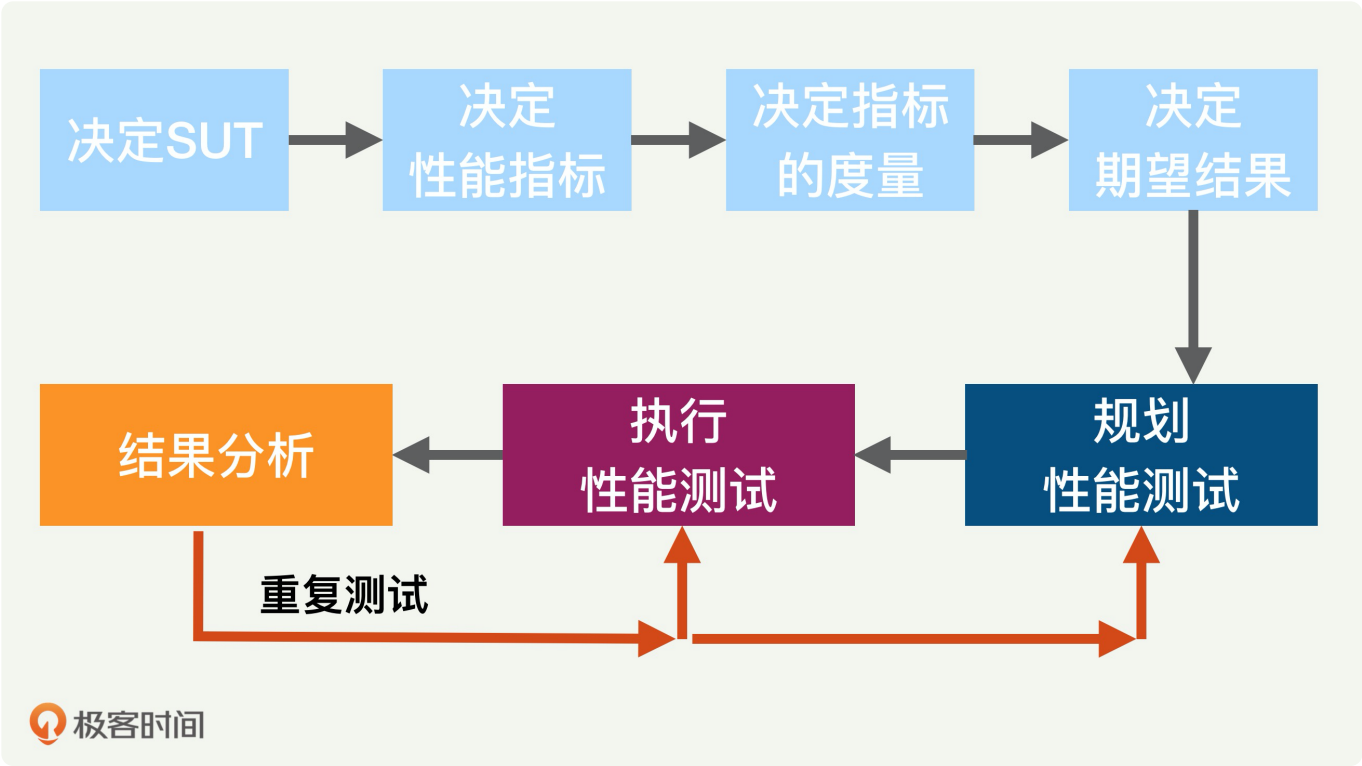


讲述：庄振运
时长 11:39　大小 10.67M

你好，我是庄振运。

上一讲我们讲了不同类型的性能测试。今天我们来讲**如何规划一个性能测试和具体的执行步骤**。在规划任何一种性能测试时，最重要的事情是搞清楚被测试的实体，也就是SUT（System Under Test），对应的性能指标和度量，以及期望的结果。在此基础上，根据测试的类型来决定和规划具体的测试步骤，然后执行测试，最后再合理地分析测试的结果。

为方便描述，我们用下图来表示整个性能测试的过程，总共七个部分。



大体上分为前后两大部分。前面四个部分分别是：决定 SUT、决定性能指标、决定指标的度量、决定期望结果。后面三个部分是性能测试的规划、测试的执行和结果分析。这三个部分根据测试的结果或许需要重复多次。

搞清楚测试对象

性能测试当然首先要搞清楚测试对象。但说起来有意思，我看到过很多做性能测试的人，对“什么才是他的测试对象”这个问题糊里糊涂的。经常碰到的情况是，有些人做了一大堆测试，但后来发现搞错测试对象了，所以大量的工作白做了。

为什么会导致这样的误会呢？

因为一个被测试的系统往往是复杂的，包含多个子系统和模块。如果对测试的类型和规划没有搞透彻，就很容易搞不清真正的测试对象。

测试的对象一般叫 SUT（System Under Test），它可以是一段代码、一个模块、一个子系统或者一个整个的系统。比如要测试一个在线互联网服务的性能，那么这整个系统，包括软件、硬件和网络，都算是 SUT。再比如，SUT 也可以是一个子系统，比如运行在某台服务器上的一个进程。

搞清楚 SUT 的重要之处，是让测试做到有的放矢。除了 SUT 本身，其他所有的模块和构件在整个性能测试的过程中都不能有任何性能瓶颈。

比如测试一个在线服务，那么所使用的负载和流量模块就不能成为瓶颈。如果这一点不能得到保证，那么性能测试得出的数据和结论就是不正确的。拿上面的互联网服务来举个例子，如果性能测试中负责产生负载流量的模块成了瓶颈，一秒钟只能发出一千个请求，那么你测出的吞吐量最多也就每秒一千请求。这样的结果和结论显然是不对的。

决定测试的性能指标

搞清楚测试对象 SUT 之后，下一步就是决定具体的性能指标。

对一个面向终端客户的 SUT 而言，一般就是和客户直接相关的性能指标，比如客户（端到端的）服务延迟。如果 SUT 是系统中的某个模块，那么测试的指标有可能是资源的使用率，比如 CPU 或者内存使用率。

平时用的最多的性能指标有三个，就是服务响应时间，服务吞吐量和资源利用率。这三个指标各有侧重，分别对应了终端客户、业务平台以及容量系统。通常，响应时间是用户关注的指标，吞吐量是业务关注的指标，资源利用率是系统关注的指标。

决定测试指标的度量

决定性能指标后，还需要更加具体到统计上的度量。比如你关注的是平均值，还是百分位数（例如 P99）；也或许是某个置信区间的大小。

举例来说，对服务响应时间延迟的指标而言，一般需要同时考虑平均值、中位数和几个高端的百分位数，比如 99 百分位。

决定性能测试的期望结果

SUT 和性能指标都确定了，那么下一步就是决定**我们期望从测试中得出什么样的结论**，比如是为了确认 SUT 的性能满足一定的指标呢，还是只希望获取一些性能数据做参考。搞清楚了测试的期望结果，才能决定什么样的测试结果是可以接受的，什么样是不能接受的。

假设 SUT 是一个互联网服务，测试指标是端到端的平均服务延迟。我们或许已经知道可以接受的平均服务延迟的最大值，比如 500 毫秒。如果性能测试测出的结果显示平均服务延

迟是 600 毫秒，那么这个测试结果显然是负面的，就是被测互联网服务不够好，不能接受。如果只是想获取性能数据，那么这个 600 毫秒就是测试结果。

再举几个更复杂一点的测试期望结果的例子：

当 2000 个用户同时访问网站时，所有客户的 P99 响应时间不超过 2 秒；

测试应用程序崩溃前可以处理的最大并行用户数；

测试同时读取 / 写入 500 条记录的数据库执行时间；

在峰值负载条件下，检查应用程序和数据库服务器的 CPU 和内存使用情况；

验证应用程序在不同负载条件下，比如较低、正常、中等和重载条件下的响应时间。

性能测试的规划

一个成功的性能测试离不开具体的规划，比如如下的几个方面的重要内容，包括负载流量的特征、负载如何注入、测试的数据、黑盒还是白盒测试、测试的工具、测试的环境等，我们逐一说明。

负载流量的特征和我们上一讲讲过的测试类型直接相关。首先我们需要决定是用真正的生产环境的负载还是仿真的负载。

那么**负载如何注入**呢？负载流量即使已经确定用真正的生产负载，还需要继续决定几个问题：

是用实时的流量呢，还是用过去捕捉的流量来重新注入？

流量的大小，是完全模拟生产环境呢，还是加大负载。

如果不使用实时生产流量，那么如何注入呢？

很多情况下，直接采用开源的工具就够了，但有些情况下需要自己开发或者对开源工具进行二次开发。

很多负载需要操作数据，比如数据库查询。所以，我们就需要决定**测试的数据**，是用真正的用户数据还是仿真的数据。

测试选**黑盒**还是**白盒**呢？黑盒就是不改变 SUT，完全做被动观察。白盒就是允许改变 SUT，比如在程序中间输出更多的性能日志信息等。白盒的问题就是改变了 SUT 的行为，可能导致最终得到的数据失真。

市场上有各种各样的性能**测试工具**，比如 JMeter，但是选择什么样的测试工具将取决于许多因素，例如支持的协议类型、许可证成本、硬件要求、平台支持等。我们后面会有一讲专门讲各种工具。

配置一个合适的**测试环境**很重要，理想情况下，应该尽量用与生产平台相同的硬件、路由器配置、网络，甚至是网络背景流量等。不过值得说明的是，有时候我们会特意选取和生产环境不同的测试环境，比如当我们希望提高可重复性，降低测试环境的噪音；那么我们就会选取一个单独的不受干扰的环境来测试。

性能测试的执行

测试规划完毕后就是执行了，这个过程相对简单。

但是需要强调的是，**测试结果的可重复性非常重要**。性能测试和性能优化很多情况下是一个长期的行为，所以需要固定测试性能指标、测试负载、测试环境，这样才能客观反映性能的实际情况，也能展现出优化的效果。

很多性能测试比较复杂，所以不要期望一次测试就能成功让整个测试环境工作。经常需要实验好几次才能真正让整个测试环境搭配成功。

所以，复杂的性能测试需要多次迭代执行，一般有以下几种方式迭代：

1. 分步进行：把复杂的测试验证过程分成几步，一次验证一步，最后一步是整个完整的测试。这样的好处是每一步的问题都可以及早暴露，快速解决。
2. 先短时间测试，再长时间测试：有些测试需要执行很长时间，比如一周。如果一周后才发现测试过程有错误，那就浪费了一周时间。所以，为了避免浪费时间，会先进行短期测试，比如半小时。然后分析结果，来发现其中的问题。这样可以比较快速地纠正测试中的错误。
3. 模拟测试：在实际使用负载测试之前，先执行简单的负载测试以检查各种工具的正确性。

分析测试结果

测试完毕，就需要分析测试结果了。

如果对一次测试的结果我们不满意，我们就需要重新回到以前的步骤上，或者重新执行测试的步骤，或者重新规划测试的方法。

根据我的经验，几乎所有的性能测试，就算是看起来非常简单直白的测试，都需要反复进行多次，才能达到满意的效果。 所以如果发生这样的情况，你千万不要气馁。

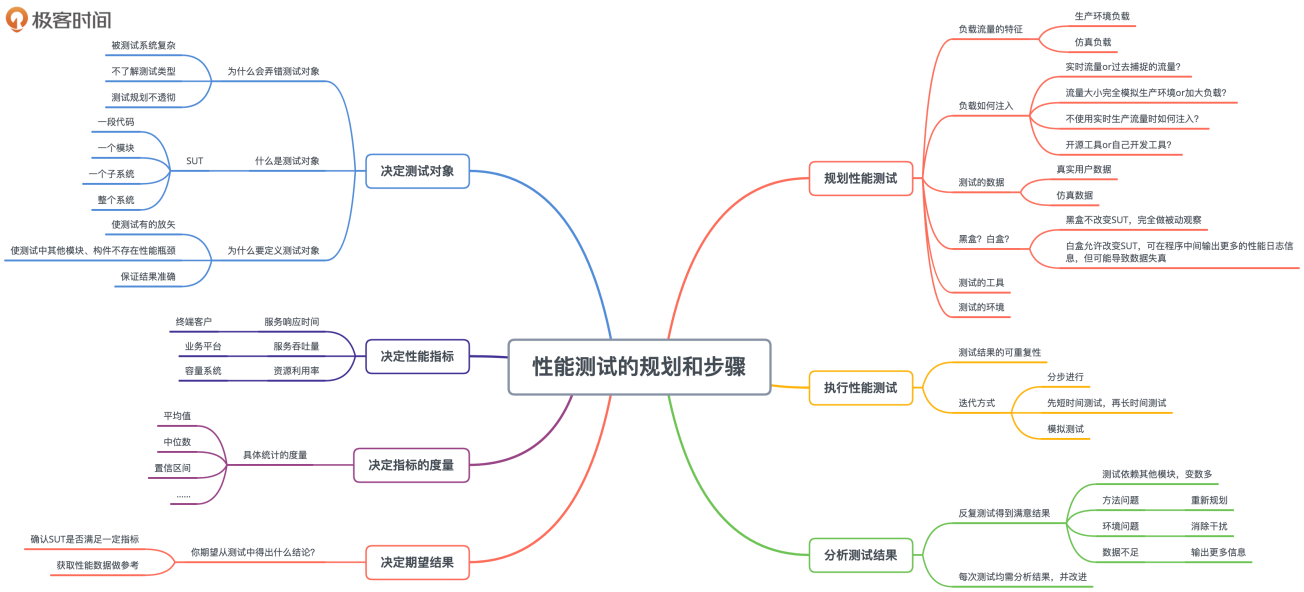
为什么性能测试不容易一蹴而就呢？

这是因为任何测试，其实都依赖于很多其他模块，比如流量的产生、数据的注入、环境的搭建、干扰的排除、数据的收集、结果的稳定等，这些模块都不简单。所以寄希望于“毕其功于一役”，一次就完美地规划和执行一个测试，几乎是不可能的。

每一次测试完毕，我们都要认真分析一下结果，如果不满意，就需要看看如何改进。如果是测试方法不对，就需要重新规划。如果是环境不稳定，有干扰，那么就需要考虑如何消除干扰，净化测试环境。如果数据的收集不够多，就需要从测试模块中输出更多的信息。

总结

郑板桥在他的七言绝句《竹石》中说：“咬定青山不放松，立根原在破岩中“，赞扬竹子目标明确，基础扎实，而且百折不挠。



我们做性能测试也是如此。只有条理分明，目标清楚，目的明确，规划仔细，执行得力，才能“千磨万击还坚劲，任尔东西南北风“。

这样的测试或许需要执行很多次，因为经常需要调整测试的方法，但不达目的，我们决不罢休。

思考题

假设你需要重复做一种性能测试，但是你发现每次的测试结果都很不一样，你可以想一想，会有哪些原因呢？举几个例子，或许 SUT 服务器上面还在跑其他程序，也或许注入的负载流量不稳定，还有其他因素吗？

欢迎你在留言区分享自己的思考，与我和其他同学一起讨论，也欢迎你把文章分享给自己的朋友。

© 版权归极客邦科技所有，未经许可不得传播售卖。 页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇　09 | 性能测试的种类：如何快准狠地抓住一个测试的本质？

精选留言 (1)

写留言

han

2019-12-18

性能测试是个大的话题，这里的每个小节，展开都会有很多细节
- 测试设计(test design)
- 测试工具选择和使用
- 测试环境搭建
- 测试数据准备...

展开

1